Module 7: Data Wrangling with Pandas

CPE311 Computational Thinking with Python

Submitted by: Marquez, Keith Leigh Zhen R.

Performed on: 03/20/2024

Submitted on: 03/20/2024

Submitted to: Engr. Roman M. Richard

7.1 Supplementary Activity

Using the datasets provided, perform the following exercises:

Exercise 1

We want to look at data for the Facebook, Apple, Amazon, Netflix, and Google (FAANG) stocks, but we were given each as a separate CSV file. Combine them into a single file and store the dataframe of the FAANG data as faang for the rest of the exercises:

1. Read each file in.
2. Add a column to each dataframe, called ticker, indicating the ticker symbol it is for (Apple's is AAPL, for example). This is how you look up a stock. Each file's name is also the ticker symbol, so be sure to capitalize it.
3. Append them together into a single dataframe.
4. Save the result in a CSV file called faang.csv.

1. Read each file in.

```
import pandas as pd
aapl = pd.read_csv('/content/aapl.csv')
```

```
import pandas as pd
amzn = pd.read_csv('/content/amzn.csv')
```

```
import pandas as pd
fb = pd.read_csv('/content/fb.csv')
```

```
import pandas as pd
goog = pd.read_csv('/content/goog.csv')
```

```
import pandas as pd
nflx = pd.read_csv('/content/nflx.csv')
```

2. Add a column to each dataframe, called ticker, indicating the ticker symbol it is for (Apple's is AAPL, for example). This is how you look up a stock. Each file's name is also the ticker symbol, so be sure to capitalize it.

```
aapl.loc[:, "ticker"] = "AAPL"
aapl.head()
```

	date	open	high	low	close	volume	ticker
0	2018-01-02	166.9271	169.0264	166.0442	168.9872	25555934	AAPL
1	2018-01-03	169.2521	171.2337	168.6929	168.9578	29517899	AAPL
2	2018-01-04	169.2619	170.1742	168.8106	169.7426	22434597	AAPL
3	2018-01-05	170.1448	172.0381	169.7622	171.6751	23660018	AAPL
4	2018-01-08	171.0375	172.2736	170.6255	171.0375	20567766	AAPL

Next steps: View recommended plots

```
amzn.loc[:, "ticker"] = "AMZN"
amzn.head()
```

	date	open	high	low	close	volume	ticker
0	2018-01-02	1172.00	1190.00	1170.51	1189.01	2694494	AMZN
1	2018-01-03	1188.30	1205.49	1188.30	1204.20	3108793	AMZN
2	2018-01-04	1205.00	1215.87	1204.66	1209.59	3022089	AMZN
3	2018-01-05	1217.51	1229.14	1210.00	1229.14	3544743	AMZN
4	2018-01-08	1236.00	1253.08	1232.03	1246.87	4279475	AMZN

Next steps: View recommended plots

```
fb.loc[:, "ticker"] = "FB"
fb.head()
```

	date	open	high	low	close	volume	ticker
0	2018-01-02	177.68	181.58	177.5500	181.42	18151903	FB
1	2018-01-03	181.88	184.78	181.3300	184.67	16886563	FB
2	2018-01-04	184.90	186.21	184.0996	184.33	13880896	FB
3	2018-01-05	185.59	186.90	184.9300	186.85	13574535	FB
4	2018-01-08	187.20	188.90	186.3300	188.28	17994726	FB

Next steps: View recommended plots

```
goog.loc[:, "ticker"] = "GOOG"
goog.head()
```

	date	open	high	low	close	volume	ticker
0	2018-01-02	1048.34	1066.94	1045.23	1065.00	1237564	GOOG
1	2018-01-03	1064.31	1086.29	1063.21	1082.48	1430170	GOOG
2	2018-01-04	1088.00	1093.57	1084.00	1086.40	1004605	GOOG
3	2018-01-05	1094.00	1104.25	1092.00	1102.23	1279123	GOOG
4	2018-01-08	1102.23	1111.27	1101.62	1106.94	1047603	GOOG

Next steps: View recommended plots

```
nflx.loc[:, "ticker"] = "NFLX"
nflx.head()
```

	date	open	high	low	close	volume	ticker
0	2018-01-02	196.10	201.65	195.4200	201.07	10966889	NFLX
1	2018-01-03	202.05	206.21	201.5000	205.05	8591369	NFLX
2	2018-01-04	206.20	207.05	204.0006	205.63	6029616	NFLX
3	2018-01-05	207.25	210.02	205.5900	209.99	7033240	NFLX
4	2018-01-08	210.02	212.50	208.4400	212.05	5580178	NFLX

Next steps: View recommended plots

3. Append them together into a single dataframe.

```
faang = pd.concat([aapl,amzn,fb,goog,nflx])
faang
```

	date	open	high	low	close	volume	ticker
0	2018-01-02	166.9271	169.0264	166.0442	168.9872	25555934	AAPL
1	2018-01-03	169.2521	171.2337	168.6929	168.9578	29517899	AAPL
2	2018-01-04	169.2619	170.1742	168.8106	169.7426	22434597	AAPL
3	2018-01-05	170.1448	172.0381	169.7622	171.6751	23660018	AAPL
4	2018-01-08	171.0375	172.2736	170.6255	171.0375	20567766	AAPL
...	...	...	...	...	...	...	...
246	2018-12-24	242.0000	250.6500	233.6800	233.8800	9547616	NFLX
247	2018-12-26	233.9200	254.5000	231.2300	253.6700	14402735	NFLX
248	2018-12-27	250.1100	255.5900	240.1000	255.5650	12235217	NFLX
249	2018-12-28	257.9400	261.9144	249.8000	256.0800	10987286	NFLX
250	2018-12-31	260.1600	270.1001	260.0000	267.6600	13508920	NFLX

1255 rows × 7 columns

Next steps: View recommended plots

4. Save the result in a CSV file called faang.csv.

```
faang.to_csv('/content/faang.csv', index=False)
```

Exercise 2

• With faang, use type conversion to change the date column into a datetime and the volume column into integers. Then, sort by date and ticker.

• Find the seven rows with the highest value for volume.

• Right now, the data is somewhere between long and wide format. Use melt() to make it completely long format. Hint: date and ticker are our ID variables (they uniquely identify each row). We need to melt the rest so that we don't have separate columns for open, high, low, close, and volume.

• With faang, use type conversion to change the date column into a datetime and the volume column into integers. Then, sort by date and ticker.

```
faang['date'] = pd.to_datetime(faang['date'])
faang['volume'] = faang['volume'].astype(int)
faang.dtypes
```

```
date      datetime64[ns]
open             float64
high             float64
low              float64
close            float64
volume             int64
ticker            object
dtype: object
```

```
#SORT
faang.sort_values(by=['date', 'ticker'])
faang
```

	date	open	high	low	close	volume	ticker
0	2018-01-02	166.9271	169.0264	166.0442	168.9872	25555934	AAPL
1	2018-01-03	169.2521	171.2337	168.6929	168.9578	29517899	AAPL
2	2018-01-04	169.2619	170.1742	168.8106	169.7426	22434597	AAPL
3	2018-01-05	170.1448	172.0381	169.7622	171.6751	23660018	AAPL
4	2018-01-08	171.0375	172.2736	170.6255	171.0375	20567766	AAPL
...	...	...	...	...	...	...	...
246	2018-12-24	242.0000	250.6500	233.6800	233.8800	9547616	NFLX
247	2018-12-26	233.9200	254.5000	231.2300	253.6700	14402735	NFLX
248	2018-12-27	250.1100	255.5900	240.1000	255.5650	12235217	NFLX
249	2018-12-28	257.9400	261.9144	249.8000	256.0800	10987286	NFLX
250	2018-12-31	260.1600	270.1001	260.0000	267.6600	13508920	NFLX

1255 rows × 7 columns

Next steps: View recommended plots

- **Find the seven rows with the highest value for volume.**

```
faang.nlargest(7,['volume'])
```

	date	open	high	low	close	volume	ticker
142	2018-07-26	174.8900	180.1300	173.7500	176.2600	169803668	FB
53	2018-03-20	167.4700	170.2000	161.9500	168.1500	129851768	FB
57	2018-03-26	160.8200	161.1000	149.0200	160.0600	126116634	FB
54	2018-03-21	164.8000	173.4000	163.3000	169.3900	106598834	FB
182	2018-09-21	219.0727	219.6482	215.6097	215.9768	96246748	AAPL
245	2018-12-21	156.1901	157.4845	148.9909	150.0862	95744384	AAPL
212	2018-11-02	207.9295	211.9978	203.8414	205.8755	91328654	AAPL

- **Right now, the data is somewhere between long and wide format. Use melt() to make it completely long format. Hint: date and ticker are our ID variables (they uniquely identify each row). We need to melt the rest so that we don't have separate columns for open, high, low, close, and volume.**

```
# melt()
faang.melt(id_vars = ['date','ticker'],
           value_vars = ['open','high','low','close', 'volume']
)
```

```
            date  ticker  variable         value
   0  2018-01-02    AAPL      open  1.669271e+02
   1  2018-01-03    AAPL      open  1.692521e+02
   2  2018-01-04    AAPL      open  1.692619e+02
   3  2018-01-05    AAPL      open  1.701448e+02
   4  2018-01-08    AAPL      open  1.710375e+02
 ...         ...     ...       ...           ...
6270  2018-12-24    NFLX    volume  9.547616e+06
6271  2018-12-26    NFLX    volume  1.440274e+07
6272  2018-12-27    NFLX    volume  1.223522e+07
6273  2018-12-28    NFLX    volume  1.098729e+07
6274  2018-12-31    NFLX    volume  1.350892e+07

6275 rows × 4 columns
```

Exercise 3

• Using web scraping, search for the list of the hospitals, their address and contact information. Save the list in a new csv file, hospitals.csv.

• Using the generated hospitals.csv, convert the csv file into pandas dataframe. Prepare the data using the necessary preprocessing techniques.

```
hospital_csv = pd.read_csv('/content/hospital.csv')
hospital_csv
```

```
# I dont have any idea how to do this
```

7.2 Conclusion